CONNEXIONS™

The Interoperability Report

August 1988 | *Special Issue: Protocol Testing* | Volume 2, No. 8

ConneXions - The Interoperability Report tracks current and emerging standards and technologies within the computer and communications industry.

In this issue:

ConneXions is published by Advanced Computing Environments, 480 San Antonio Road, Suite 100, Mountain View, CA 94040, USA. Phone: 415-941-3399.

ISSN 0894-5926

From the Editor

Many of the products we purchase as consumers have been through some sort of stamp-of-approval process. Examples include electrical appliances which in the United States are tested by the Underwriters Laboratory and carry a "UL Listed" sticker. Other countries have similar institutions responsible for ensuring that products conform to certain standards for functionality and safety.

In the protocol world we are mostly interested in *interoperability:* "Will this package from vendor A let me talk to a package from vendor B?" One way to find out is to do interoperability testing in a laboratory environment, but when you have a large number of implementations this can be an expensive and time-consuming exercise. A more common approach is to do *conformance testing*, that is testing each implementation against a reference system.

In this special issue, we will look at 3 such testing laboratories. The first is the *DCA Protocol Testing Laboratory* developed by Unisys. The article is by Bob Jones, previously with Unisys and now with TASC. The second article describes the *BBN Test Net* and is by Tony Michel of BBN Communications Corporation. In the OSI arena, major testing facilities are being developed by the Corporation for Open Systems (COS). COS has a set of *Conformance Test Systems* as outlined in an article by Chris Rohrer on page 21.

In the DoD world, a set of conformance test documents for MIL-STD protocols have been developed. These documents are now available for comments, see page 15 for more details.

Related to protocol testing and verfication is the ability to look closely at what goes "on the wire". There are several tools which allow you to analyze data traffic on a variety of media. Harry Saal, president of Network General describes some experiences with his company's protocol analysis tool, *The Sniffer*(TM) on page 16.

Finally, a reminder about next month's big event: *INTEROP 88: The 3rd TCP/IP Interoperability Conference & Exhibition*, will be held at the Santa Clara Convention Center and Doubletree Hotel, September 26 - 30. By now you should have received the Advance Program in the mail. Let us know if you didn't. You may register for the conference by calling 415-941-3399.

See you in Santa Clara!

The DCA Protocol Testing Laboratory

by Bob Jones

Introduction

This is a follow-up to an earlier article on the DCA Protocol Laboratory which has been under development for over four years. [Ed.: See *ConneXions*, Volume 1, No. 6. October 1987]. The laboratory provides a fully automated capability to initialize and execute functional and qualitative tests; recovery from internal and external perturbations (i.e., a flawed protocol implementation or the underlying network) for the continuation of the testing process, and a complete results reporting facility. The test system is designed to operate over the Internet or on local area networks. Some performance tests are also provided for the lower level protocols.

Testing methodology

The major goal in the development of the laboratory was to provide a capability to test the complete suite of DoD protocols (IP, TCP, FTP, SMTP and Telnet) to insure that they are in compliance with the appropriate Military Standards (MIL-STD's). While these protocols are tested separately (i.e., access to the upper and lower interfaces), there is a capability available in the laboratory to test full "stacks", where the interfaces are tightly-coupled and not well defined. This is referred to as *Vanilla Testing* and will be discussed later.

Testing across the Internet

Since the DoD protocol architecture is designed to support a network of dissimilar hosts, and DoD systems are built on a wide variety of host computers, it is impractical for the laboratory to own every machine on which an implementation is possible. Nor is it possible to install every machine in the laboratory. Therefore, the machines and protocol implementations under test (IUT) are run across the Internet or via dial-up lines to the laboratory. In order to test any single protocol, driver software must be developed and installed in the machine that hosts the IUT. That driver software is specified in documents for each protocol.

Connection integrity

Protocols are specified in terms of extended or finite state machines that define the functional characteristics of interacting communicating entities. Actual protocol implementations, on the other hand, often support multiple independent communications. For example, TCP should support several independent connections. The number of such connections, the way they compete for resources, and the mechanisms for keeping them separate are implementation specific. The testing can determine whether the independent connections interfere with each other with regard to precedence and integrity of the data flow, with each implementation having its own definition of data flow integrity. The TCP tests check that data on a connection be delivered in sequence, without alteration and without introducing spurious data from any other source or from the connection itself. In contrast, IP tests check that the data, if it is delivered at all (remember, this is datagram service), be delivered to the intended upper level interface (in most cases this will be TCP).

Laboratory Central Driver

Testing is performed under the control of a central facility called the *Laboratory Central Driver*. As Figure 1 shows, the site under test houses a remote driver that passes commands from the Central Driver to the IUT. Another driver is the Laboratory Slave Driver, which passes commands from the Central Driver to the Laboratory Reference Implementation of the protocol.

Slave drivers, Reference Implementations and Central Drivers (one for each protocol) are part of the Protocol Laboratory. The Remote Driver is implemented by the candidate under test (usually a vendor) prior to testing, and therefore not considered part of the laboratory.

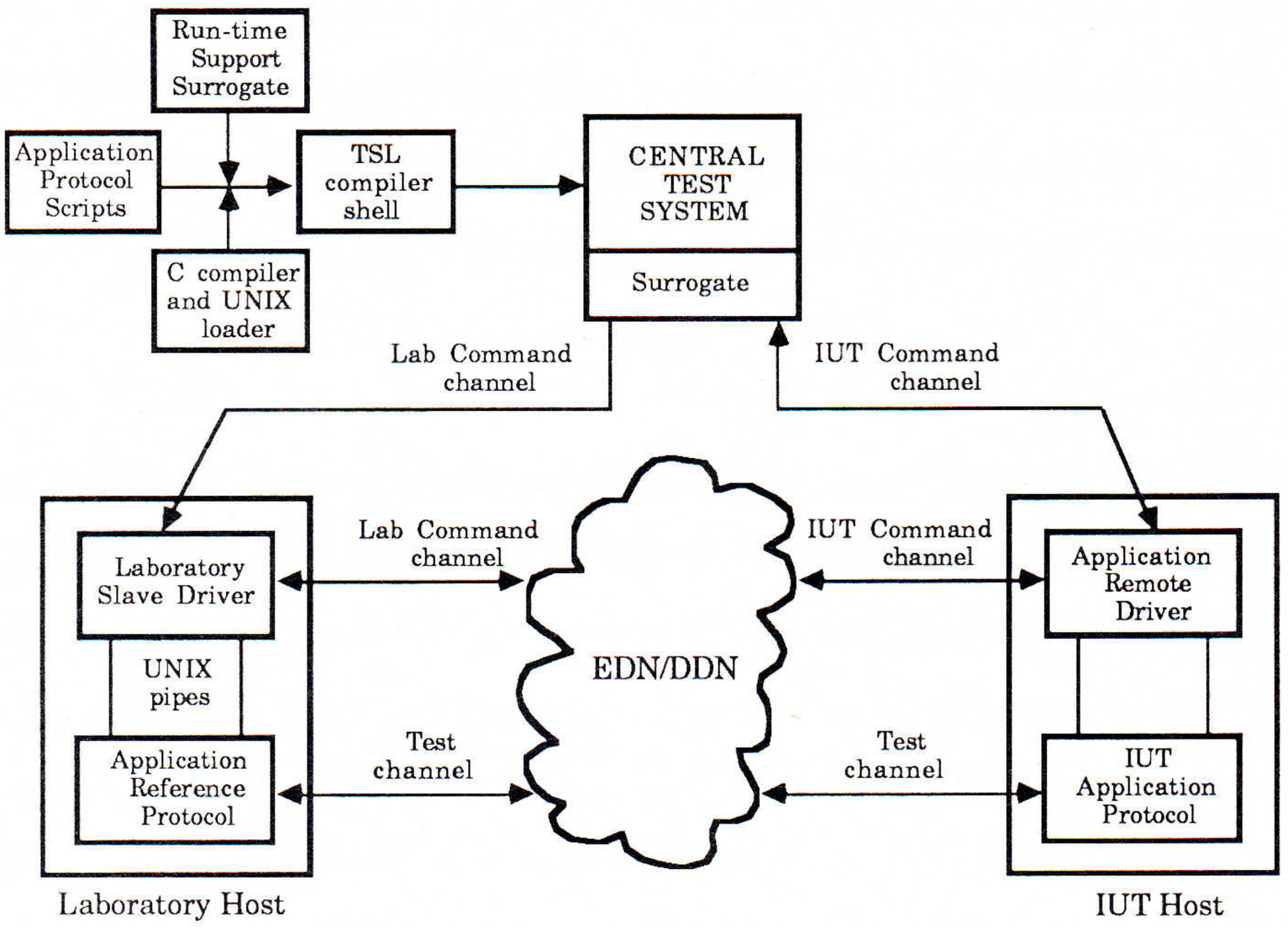


Figure 1: Architecture for testing FTP, Telnet, SMTP, TCP, and IP

The laboratory maintains comprehensive tests demonstrating that the IUT is able to carry out normal communications with the reference version of the protocols. In addition, the laboratory provides facilities to intentionally generate altered protocol exchanges (corrupted packets). This process validates whether the IUT is robust enough to receive incorrect sequences and to reject the types of improper sequences it was designed to reject, while generating the appropriate responses (i.e., error codes) to its peer.

Test Scenario Language (TSL)

The laboratory uses *TSL*, a high-order test development language that provides commands for test initialization, control and coordination, results analysis and logging, and test completion (gracefully closing a test process and all connections). The initialization statements allow the user to open command connections from the Central Driver to either the Remote or the Laboratory Slave Driver. When the command connections are established, TSL provides commands for sending protocol or driver specific commands, and for receiving the results of the command. TSL statements allow the user to analyze the response for correctness and validity. TSL-supplied conditional statements can be used with evaluation statements to direct the flow of the test through the various scripts available at execution. TSL provides a variety of logging statements that allow the scriptwriter to direct the output information to file storage media or to both an interactive display and storage media.

Associated with the TSL is a compiler and run-time support package. Figure 1 illustrates the TSL compiler processing commands into **C** statements and calling the **C** compiler for object code generation.

continued on next page

The DCA Protocol Testing Laboratory *(continued)*

The object code generated is then linked with a run-time support package that provides the procedures for implementing the TSL commands. The compilation and linkage processes have been incorporated into a **C**-shell script file so that the user sees only the equivalent of a TSL compiler.

Reference implementations

The laboratory has *reference implementations* of the complete DoD protocol suite. A reference implementation is an "instrumented" version of the specific protocol that supports all of the required functions and options in accordance with the appropriate MIL-STD. The reference implentations can corrupt packets which are used for bottom-up testing (discussed later).

Test scripts

To perform the testing, many test scripts have been developed for each protocol. The scripts are combined in various ways to create scenarios that test an IUT's ability to perform a certain set of functions. All of the test scripts are written in TSL. Following are examples of features of protocols that are tested.

Bottom-up testing

These are testing events that are introduced during the major data transmission states of connection-oriented protocols and all states of connection-less protocols. Packet arrivals are at the lower level interface of the protocol sitting above the generating entity. These include replication of segments, random deletion of segments, segment modification, contiguous deletion and delay modulation. (See Figure 2)

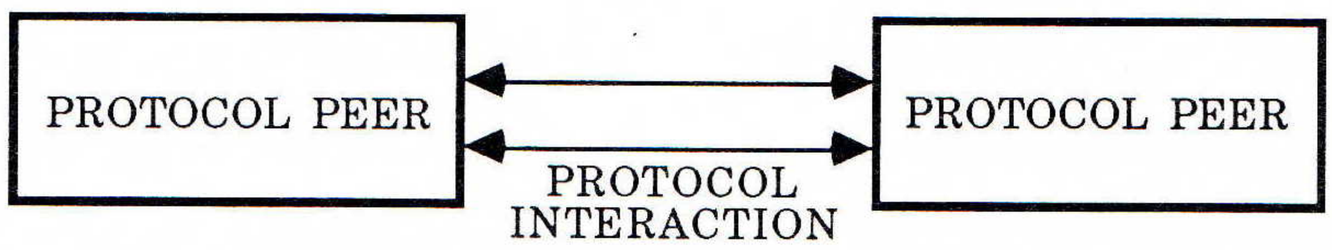


Figure 2: Bottom-Up Testing

Top-down testing

This type of testing exercises the controls available at the service (or user) interface, driving the protocol to a specific state, determining that the protocol is in the desired state, and restoring the protocol to a standard state (i.e., closed) from which further testing can be performed. (See Figures 3 and 4).

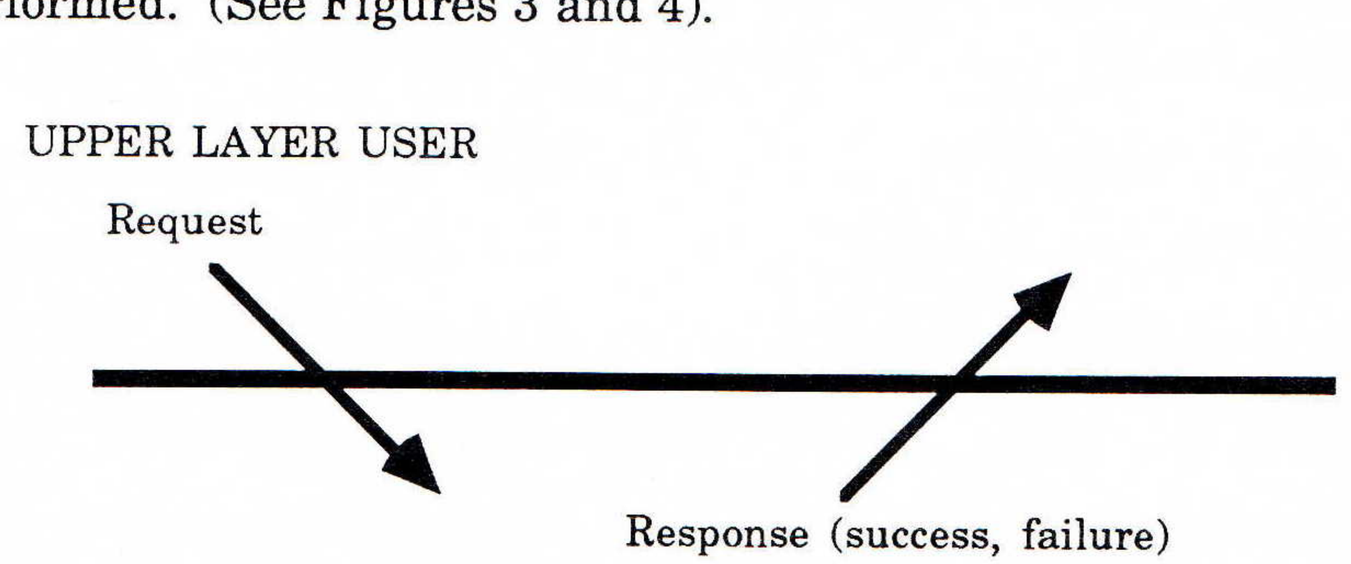


Figure 3: Upper Level Interface

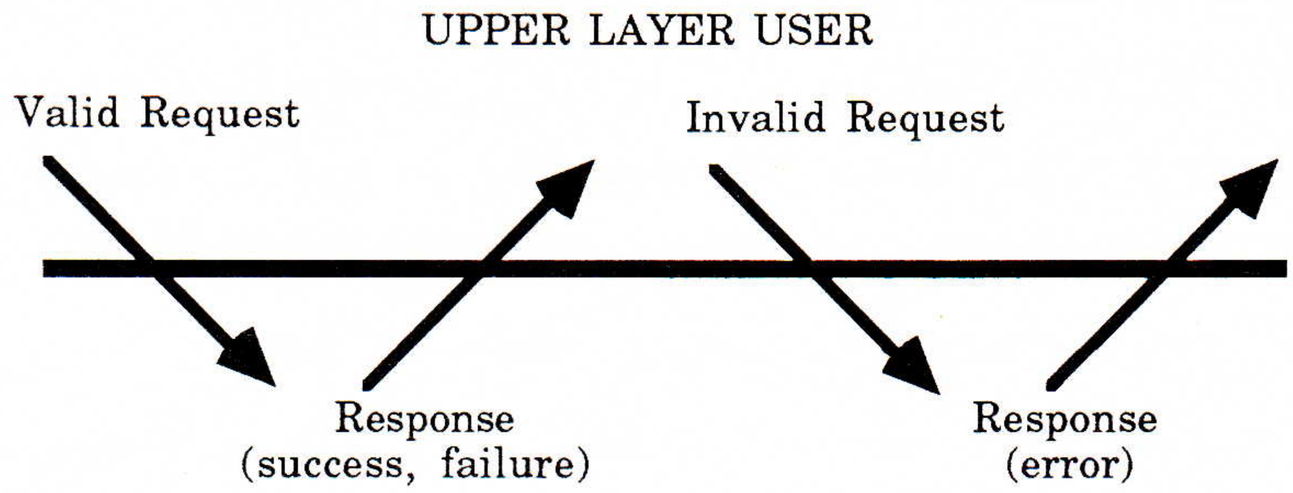


Figure 4: Upper Level Test Interface

TCP Functional testing of TCP includes the ability to do the following:

- Active open
- Passive open
- Data Transfer in Two Directions
- Graceful Close
- Abort
- Urgent Transmission in the Established state
- Push Transmission in the Established state
- Discard duplicate segments while in the Established state
- Retransmit lost segments
- Expand window when Allocate is issued
- Respond to Status
- Communicate with more than one site without mixing the connection data
- Multi-connection testing in which the behavior of the implementation is tested when the number of open connections is at or near the maximum for the implementation
- Testing of the security options including proper accomplishment of the security match during data transfer
- Validation of all legal, upper level, and timeout actions that should occur for both connection management functions and data transfer functions ("legal" and "upper level" refer to inter-process communications in the protocol suite)

IP Functional testing will validate the IP's ability to:

- Perform Local Delivery
- Communicate with a remote site without fragmentation
- Fragment and reassemble multiple fragmented datagrams
- Manage the Don't Fragment bit
- Set up Source Routing options
- Receive properly checksummed ICMP messages
- Discard flawed ICMP messages
- Perform Reassembly Timeout
- Respond to ICMP Redirect
- Generate all options and reject poorly formed options
- Accept all necessary Security and Quality of Service options

continued on next page

The DCA Protocol Testing Laboratory *(continued)*

- Handle multiple Upper Level protocols

- Respond to High Precedence traffic when there are no IP buffers.

SMTP

All three roles which SMTP performs will be tested. In general, the source initially sends a query to determine whether the user is known at the destination. If so, the message is sent; if not, the message is not sent and the sender is notified. The process is more complicated if the path passes through more than one relay (forwarder) site. At least four sites are needed to test all of the combinations of events that can occur. These four sites can be the two VAXen in the laboratory, the EDN VAX, and the facility of the implementation under test (IUT). In addition, all the legal address fields and internet address forms must be tested. Addressing will be done with legal and illegal symbolic host and network names. Both long and short messages will be used to validate the ability of the protocol to handle messages of different lengths. SMTP is a protocol that should be robust in the presence of hosts that are down, as well as in the presence of transport connection failures (error code generated in this case). Thus, reference hosts will appear to be down during part of the testing. Also, it will be possible to break transport connections during message transfer and compare the error code generations for proper reponse. The DCA Protocol Testing Laboratory provides a reference SMTP implementation which:

- Implements the complete battery of SMTP commands (i.e., all mail transactions to be used, multiple recipients to be specified and several transactions to be performed during a single SMTP session). Furthermore, it correctly executes the SMTP TURN command in which it offers to, and becomes, an SMTP server process.

- Implements a "times attempted" count for use when attempted transactions are aborted due to problems assumed to be temporary. After some number of failed tries, a notification will be mailed to the original sender.

- Implements a more sophisticated set of actions to take in the face of "non-OK" replies (previous feature). The original code provided for the sender merely to abort the session if a non-OK reply code was received.

- Implements the ability to recognize valid forward and return mail paths. The original code pretty much assumed that anything it received was in the proper format--with potentially weird results.

- Implements the ability to process all syntactically correct forms of forward and return paths. The original code could not handle host names in dotnum or #number format. Neither could it handle quoted strings or escape characters.

- Incorporates a check of the host name a sender announces during the HELO command to that associated with its IP address, as a protection against spoofing.

FTP

Testing of FTP will include the capability of the user and server FTP to move files between sender and destination sites. FTP is not a full duplex protocol; however, storage and retrieval functions will be tested. The laboratory will validate the ability of the FTP to respond to peer-generated error messages on the command channel. All data types and file structures will be tested (i.e., ASCII, EBCDIC; Block, Stream, File). The laboratory will vailidate that: (1) Files are received completely and contain the correct information; (2) Where there is a data translation or a binary file transmission, the transmission was correctly carried out (this is effectively accomplished by TCP); and (3) When the checkpoint feature is implemented, a recovery can be made. (Transport failures will be introduced at various points in different transmissions to validate this functionality). The laboratory will exercise the implementation under test (IUT) in the roles of control, sender, and recipient entities. Even if the IUT is not designed to play all three roles, all combinations of roles for the IUT will be exercised, with the laboratory playing the remaining roles.

The following user FTP commands were added:

ACCT: Send an arbitrary account to a foreign server
ALLO: Have the foreign server pre-allocate file space
PASV: Request foreign server to perform passive open
PASS: Send the password to the foreign server
PORT: Assign new data port for the foreign server
REIN: Request foreign server to re-initialize the connection
RETR: Specifically request the foreign server to send the file
SITE: Request any site-specific information
STAT: Request foreign server's status
STOR: Specifically request the foreign server to store the file

The following file transformation capabilities were added:

BLOCK: Send data according to the FTP-defined Block format
COMPRESSED: Send data using the FTP-defined Compression
RECORD STRUCTURE: Use FTP Record Structure format

The following third party commands were added:

CLOSE_HOST: Close the specified host connection
DIE: Close all open host connection
SWITCH_HOST: Change host contexts

(Note: With third party commands, the tester can cause three-party file transfers on IUTs that implement the PASV command).

Telnet

The laboratory makes available the ability to test the virtual terminal capability to support default options, connection management commands, and the transfer of ASCII text. The facility will test both Server and User versions of Telnet, singly and jointly. When tested singly, the laboratory will play the complement role. When the IUT plays both roles, the laboratory will drive the User interface and process at the Server site. The laboratory will drive both Telnet entities through the negotiation of all options. An IUT is not required to accept any option or to request any option; it is required only to correctly respond to default options.

continued on next page

The DCA Protocol Testing Laboratory *(continued)*

Although the laboratory reference implementation will respond correctly to "any option" and support all options, the IUT is, in fact, exempt from this requirement. Any response to an option not implemented should be stringed text (i.e., "option not implemented").

- The User now comes up in local echo and waits for a Go-Ahead before sending data. This is the default network virtual terminal setting.

- The user interface will allow the tester to send all of the Telnet commands.

- User Telnet will monitor the data received from the tester to see whether an option negotiation sequence is starting. If so, User Telnet will quench the negotiation at the appropriate time.

- The Server now comes up expecting not to echo, and the remote User waits for a Go-Ahead.

- The Server now talks directly to a remote driver, and there is no login sequence required.

- The commands received by the Server are also passed to the remote driver. The Server expects the commands from the remote driver interface to be in Telnet command format.

Concept of operation

The laboratory concept of operation calls for an IUT to carry out correct information exchanges with reference implementations. A complete test is the execution of a number of predetermined scenarios. Each scenario is a set of prescribed actions carried out to test a particular protocol feature. To help assure reproducible results, each implementation (i.e., reference/IUT) is controlled through its upper level interface by a driver executing a distinct script. In effect, the driver is a script interpreter. New scripts can be constructed as needed to test new protocol features and/or to produce a different level of testing. Scripts are defined by a script language. The script (or scenario) description language is designed to support a full range of command and interpretive functions. The scenario language is written in **C**, and can support the development of remote drivers by implementers who are required to submit to qualification testing under the auspices of the laboratory.

Vanilla testing

It was believed that some vendors who would implement the DoD protocol suite, would in fact, develop "stacks" of protocols, i.e., there would be no definite interface between the layers. A test architecture was put in place that would allow these stacks to be tested but because of the nature of such testing and variety of possible stack combinations, this test system is much less automated than the driver-oriented test system. The reporting process is also less formatted. Figures 5 and 6 show the layout of this architecture. The problem with this type of testing is that it requires the services of an operator continuously and the testing could go on for several days - just for one set of test sequences. This can get very expensive.

The capability exists to test tightly-coupled implementations of TCP/IP and this combination testing is fully automated under the described TSL system.

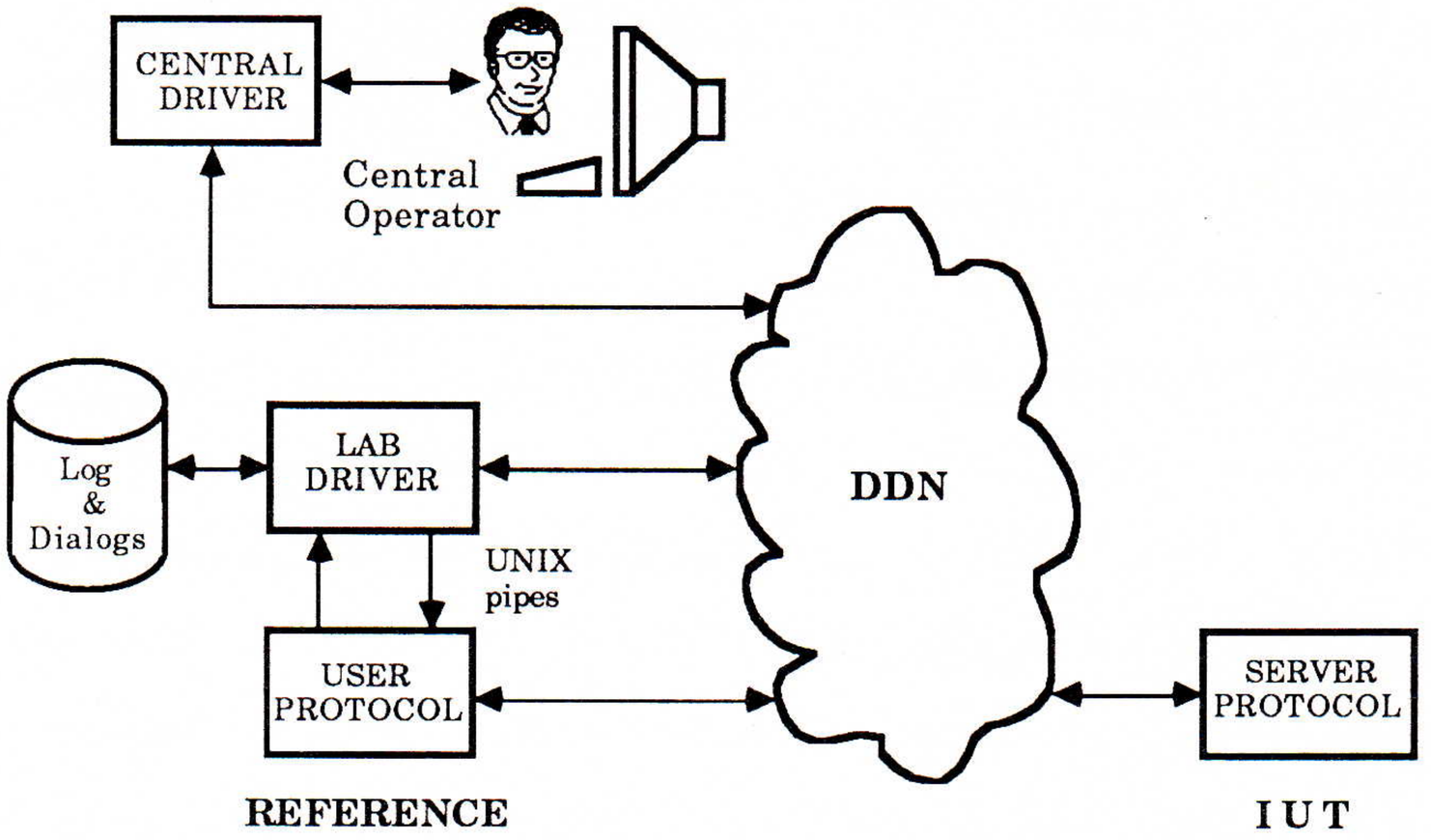


Figure 5: Vanilla Server ULP Testing Architecture

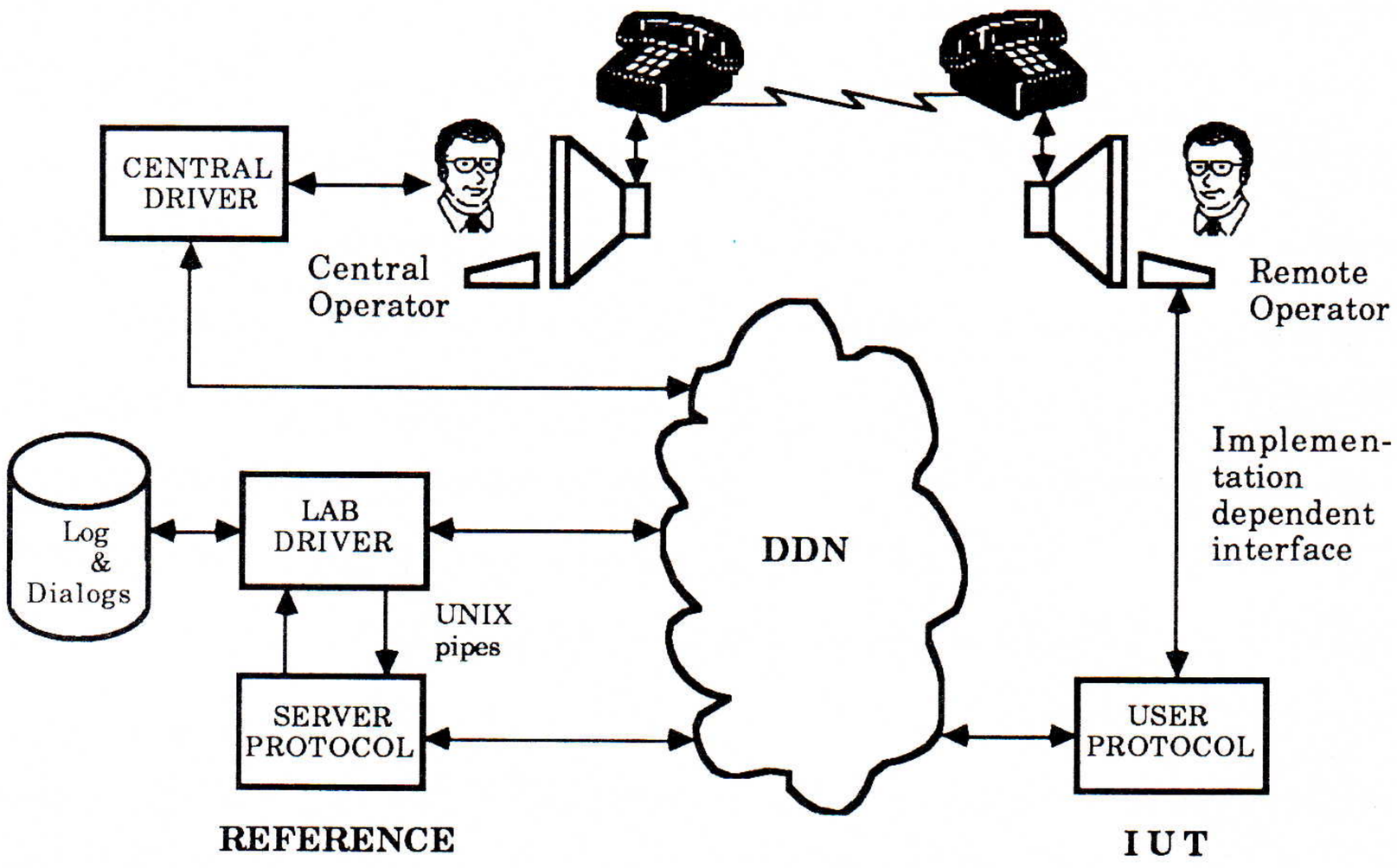


Figure 6: Vanilla User ULP Testing Architecture

continued on next page

The DCA Protocol Testing Laboratory *(continued)*

Other laboratory capabilities

In addition to the testing of existing DoD protocols, the laboratory also serves other purposes. These include:

- Serve as a resource for developing new protocols
- Test new, modified and enhanced protocols (i.e., ISO/OSI)
- Support standards efforts
- Support Internet engineering and research
- Support configuration control of networking protocols

Summary

As with any testing of this kind, there are peculiarities with each protocol test process. In other words, each protocol presents some unique functional characteristic that may or may not be testable. Where such a case exists, it will be specifically referred to in the test for that specific protocol. One example is the *Page Mode* in FTP. UNIX does not implement this mode at all, and the Military Standard is very vague about its definition. There are rare instances where a particular test process may not have been implemented because of such vagaries.

A report from Bolt, Beranek and Newman (BBN) states: "At BBN we have a number of tools to monitor the long-range performance of the Internet. Thorough reports give us detailed information; we collect statistics on the number of datagrams between each source and destination and host. In addition, we measure a wide range of parameters in Arpanet and MILNET. These include detailed throughput statistics, statistics about the end-to-end traffic and about the store-and-forward traffic. But even with all these tools (and others) at our disposal, we are stopped at the host. There we find TCP/IP implementations written by many different people and containing *subtle* differences in interpretation that could lead to major problems." This statement makes the point very well.

For more information on the DCA Protocol Testing Laboratory, contact John Swanson at Unisys in McLean, VA at 703-847-3467.

TCP/IP Reports available

Infonetics, Inc., and Advanced Computing Environments have produced 2 in-depth market analysis reports on TCP/IP to help you make the right decisions concerning your internetworking strategy:

- *TCP/IP: The Market View 1987-1990* includes a TCP/IP Market Size and Forecast as well as an overview of the top vendors and their products.

- *TCP/IP: The User Perspective 1988* furnishes specific information on TCP/IP users and interprets the strategic significance of that information to you. The report is based upon interviews with 300 users of various TCP/IP products.

To order these reports or for more information contact Joe Seidler, Infonetics, Inc. at 408-746-2500.

Protocol Testing Lab at BBN Communications

by Anthony X. Michel, BBN Communications Corp.

Introduction

BBN Communications Corporation has recently established a testing laboratory for assisting prospective users of data networks in their efforts to qualify, test, and exercise their networking hardware and software before it is released into an operational network. The laboratory, called *Test Net*, was originally built to support the needs of our commercial customers using DECNet and IBM SNA architectures. As the number of commercial vendors of TCP/IP products has grown, we have seen a need for the same testing service in this realm, and have extended our capabilities accordingly. Test Net was started because we found that if a vendor had the right testing environment, many of the problems usually uncovered after actual system integration could be diagnosed before final delivery of the product to the customer. Unfortunately, the kind of testing environment needed to uncover these problems was not readily available to most vendors until we began to offer Test Net.

Backbone

Test Net provides a realistic environment in which systems may be configured, tested, staged and prepared for use on the Defense Data Network (DDN) and other networks. The backbone of Test Net is made up of BBN C/30 packet switches. These are the same packet switches used in the DDN, using the same software as deployed in the DDN. Over the past several years the DDN has seen tremendous growth. As the DDN has grown, so has the number and variety of hosts trying to connect to it. We have several TCP/IP hosts on Test Net that are typical of some DDN hosts, and these are available for interoperability testing. This service is especially valuable to DDN vendors, since the the MILNET portion of the DDN is generally unavailable for testing purposes, and sponsorship for and connection to the Arpanet can be a long and involved process.

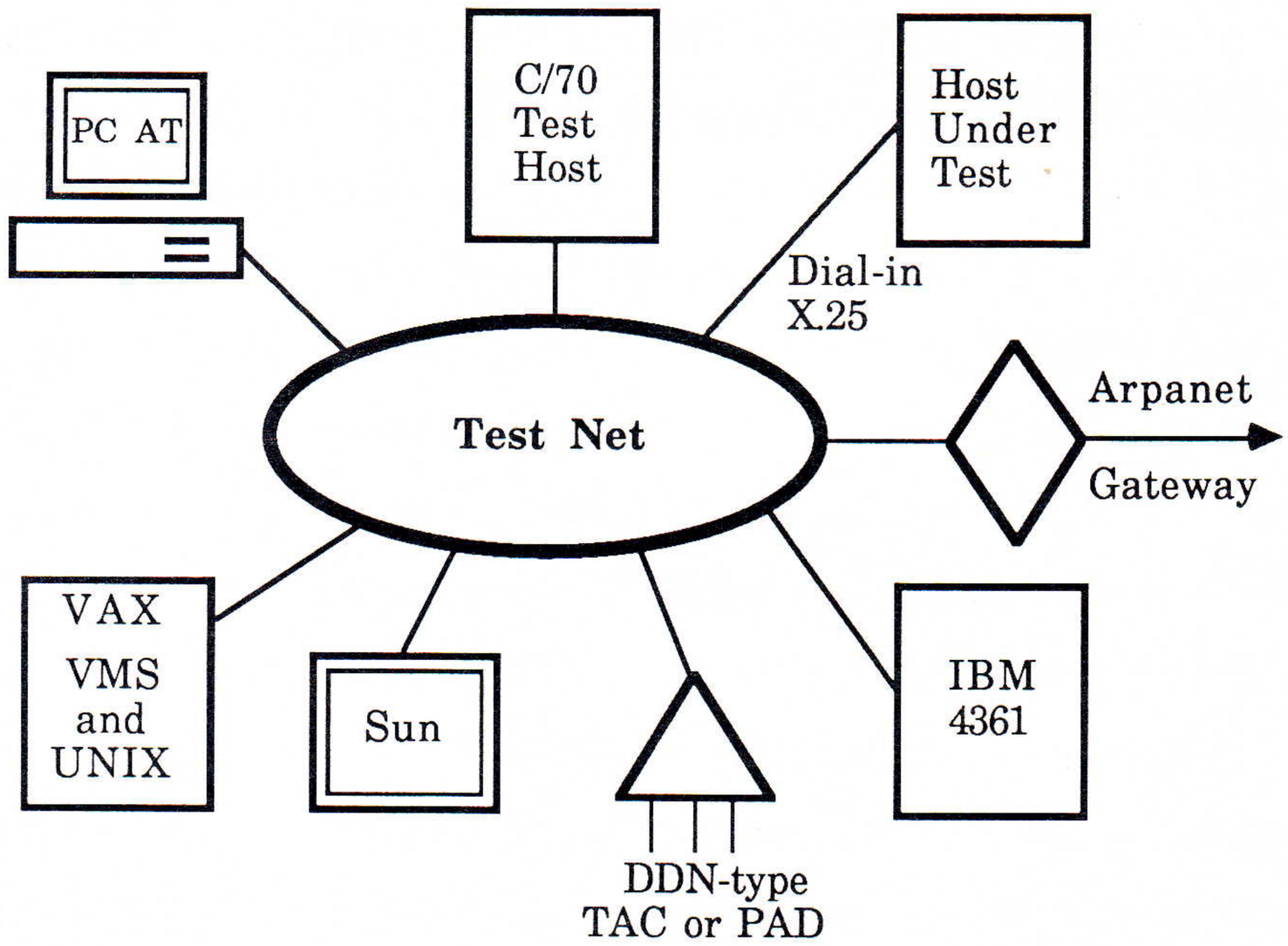


Figure 1: Test Net environment

continued on next page

Testing Laboratory at BBN *(continued)*

Our testing philosophy is pragmatic. Hosts are made to do realistic tasks under realistic operating conditions. To verify the operation of a file transfer application, for instance, we make a host actually move a useful file across an internetwork, to show that all levels of the host software perform effectively and efficiently.

Phased approach

Because some hosts initially use unproven software and hardware implementations, we employ a phased approach to testing. We begin with isolation testing, in an environment where the hosts can do no damage to operational systems. As we develop confidence in the benign nature of the Host Under Test (HUT), we allow it access to increasing levels of the internetwork. It is important to note that Test Net does *not* provide ongoing Internet access. Host testing on Test Net is restricted to the Test Net evironment except during short, specific, supervised test periods.

A standard skeleton test plan for X.25, TCP/IP and higher-level protocol testing is available, and this test plan can be customized to evolve into a function and performance test plan for candidate hosts. Our testing procedures involve some automated and autonomous operation, where HUT personnel can experiment and explore the operation of their system independently. We provide a range of operational and engineering support to assist HUT personnel in problem solving and interpreting the results of the tests.

In the future we expect to be accredited by the National Bureau of Standards as a National Voluntary Laboratory and we hope to be able to use the standard TCP/IP verification suite that was developed by the Defense Communications Agency (DCA) for this purpose. [See page 15]. In the longer term, we hope to use the test tools we are building, based on our Sun workstation, to do comparable testing on the GOSIP protocol architecture.

X.25 and Higher-level protocol testing

Our X.25 capabilities use both the DCA-developed DDN X.25 qualification test suite (based on a Tekelec datascope) and an evolving set of tools we have built into the C/30 packet switch. These tools allow us to monitor the X.25 interface on a local or remote packet switch.

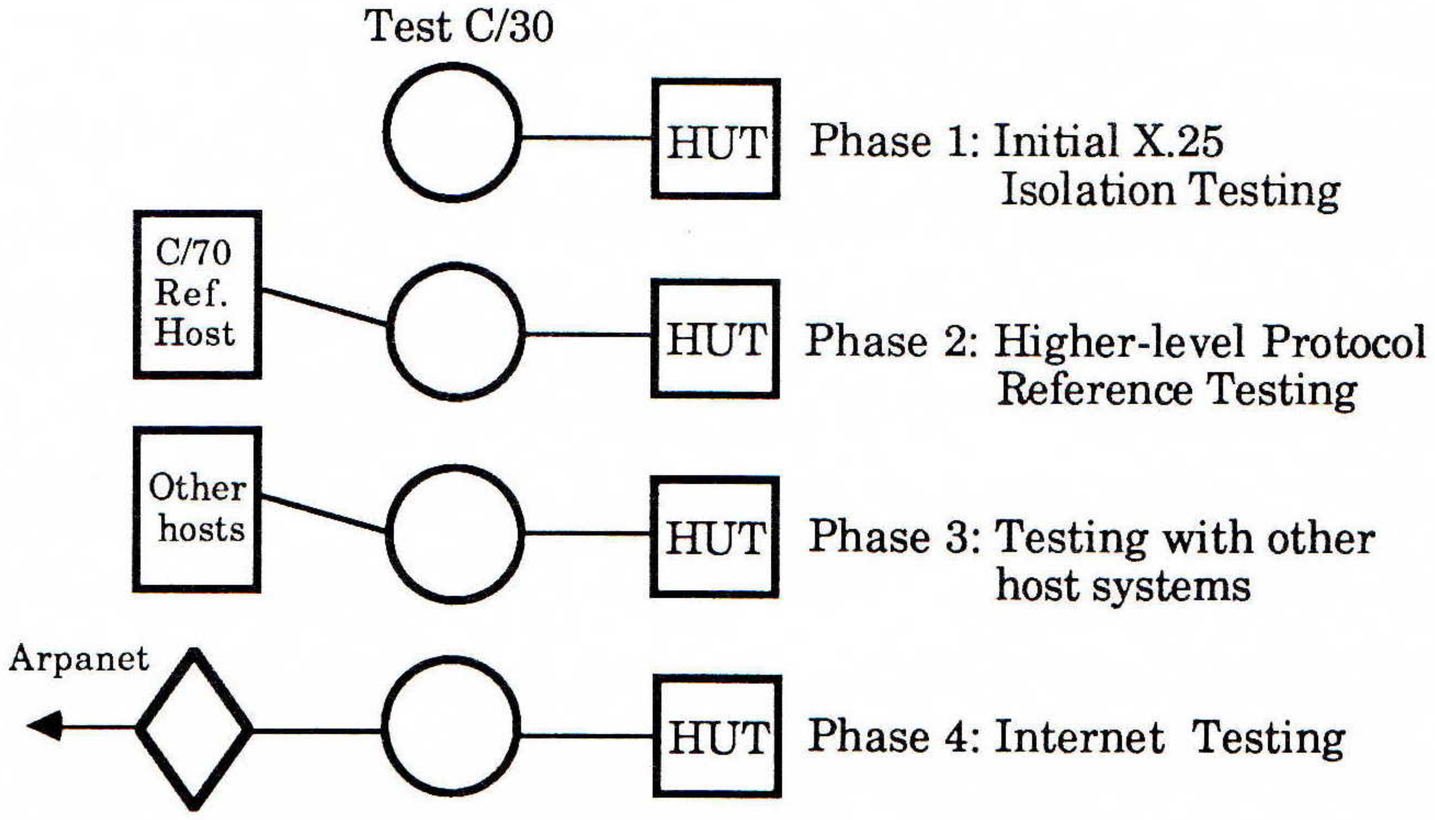


Figure 2: Test Net methodology

Our testing of higher level protocols is focused on the BBN C/70 reference test host in our Test Net test cell. The C/70 is extensively instrumented and allows us to generate detailed history "traces" of a TCP connection, and deduce the behavior and even flaws in a host under test. The C/70 also incorporates a full range of the basic DoD protocol suite, and is the basis of our initial testing of Telnet, FTP, and SMTP (and electronic-mail applications). Beyond testing with the reference C/70 we have a variety of other hosts, such as VAX, Sun and IBM (VM) to test against.

Testing an Internet

As previously mentioned, our testing methodology is a phased approach (see Figure 2). We begin with the HUT in an isolated environment, and expand the environment as we gain confidence in the host's behavior. Phase 1 of the testing is standalone X.25 testing to verify the benign host behavior; in other words, to show that the host does no damage to the operational system. As part of the Phase 1 testing, we can also help the vendor practice for the DDN X.25 qualification test using the DCA software and a Tekelec datascope. In Phase 2 of the testing, the HUT is tested against a reference host (BBN C/70) to verify the basic function of TCP, FTP, Telnet, and SMTP. Isolated function and throughput tests are performed using FTP transfer rate and Telnet Option Conformance tests. Phase 3 is testing against other typical DDN systems, like IBM, Sun, and VAX, in the Test Net environment. The final phase of the testing, Phase 4, is internetwork testing through gateways, testing under stress, and testing against real production systems. The Test Net is used at all phases of integration, testing, and operation of a data system in a TCP/IP network.

Accessing Test Net

Test Net users may access the Test Net in three ways, as shown in Figure 3:

- Dial-in X.25 host connection for initial or occasional service,
- Leased full period X.25 host connection for frequent use, and
- On-site C/30 packet switch for multiple on-site hosts high, intra-site load tests, and more flexibility.

Local access to the network is available in major U.S. cities, with dial-in access available throughout North America. In special cases, testing of a host in the BBN laboratory or on-site testing by a BBN engineer can be arranged.

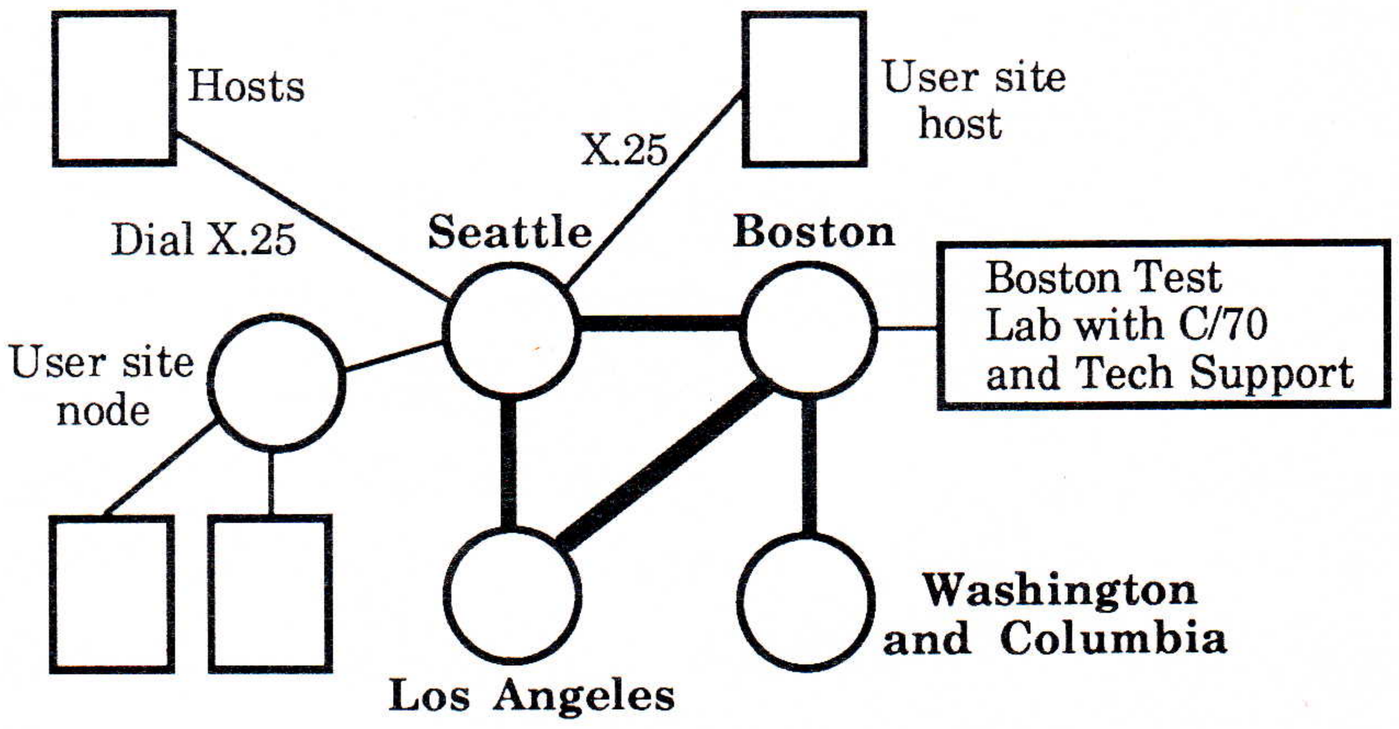


Figure 3: Test Net access options

continued on next page

Testing Laboratory at BBN *(continued)*

Uses of Test Net

Access to Test Net can be very useful to vendors in several ways. For example, Test Net can be used for qualifying new protocol implementations. New implementations generally need some polishing to ensure they do not damage the Internet and that they do use network resources efficiently. Test Net can also be used for staging new hosts before deployment to the Internet. As hosts are added to the Internet, generally each installation needs some system engineering support to make its initial connection, even working with qualified software. Test Net can provide a way to stage the installation, and BBN can provide engineering help if necessary. Regression testing of software is also possible via the Test Net connection. Each time network or host software is changed, it is generally necessary to test samples to ensure the new software works effectively. The software can be tested using the Test Net environment.

Integration and Subscriber Support

Integration support is also a part of the Test Net services. At BBN, we feel it is important to support the integration of new systems into the Internet. Usually individual host systems' personnel do not have the interest or ability to do this job well. In addition, network subscribers, both application users and system administrators, need help with understanding application behavior, troubleshooting, and planning and developing improved applications. Test Net can provide help, since our engineers have extensive experience in these areas. This Test Net service can be used by new DDN subscribers.

Experience

Since BBN began offering Test Net services nearly a year ago, we have gained alot of experience in the operation of a protocol testing laboratory. We have had several major vendors of TCP/IP products as our clients, and some of these clients, as their initial testing with Test Net has finished, have added their hosts to the Test Net or have purchased packet switches and have actually become part of the Test Net. All of our clients have found Test Net a very useful tool in testing their protocol implementations and preparing their systems for integration onto the DDN. Because our testing approach is a pragmatic one and we do not try to test every possible parameter of each protocol, we do *not* certify software for the DDN or for the Internet. However, we feel that each client, after using Test Net, can be confident that their hardware and software will work well in a real network environment.

For more information on Test Net or the services we offer, contact:

Ms. Lori Silber
Manager, Government Professional Services
BBN Communications Corporation
8000 Westpark Drive, 6th Floor
McLean, VA 22102
703-848-4855 or lsilber@cct.bbn.com.

ANTHONY X. MICHEL is Manager of the Systems Engineering Group at BBN Communications Corporation. He has been involved in packet switched networks since the inception of the Arpanet in 1969, and has lead a number of hardware and software development efforts for the DDN and related networks. Mr. Michel's interests center on the architecture of the packet switch and network access devices. At present he is involved with the systems engineering aspects of subscriber connection to DDN.

Conformance Test Docs available for comment

The Defense Communications Engineering Center (DCEC) Protocol Laboratory invites comments from the Internet community on the following draft documents. These documents outline the conformance tests developed for the DoD MIL-STD protocols. The comments can be about any aspect of the document, but in particular, opinions are requested on the accuracy and completeness of the test procedures. The documents are unrestricted so comments will be accepted from the entire Internet community. The documents are:

1. DCEC Protocol Laboratory Internet Protocol (IP)
   MIL-STD-1777 Certification Tests Index, DRAFT

2. DCEC Protocol Laboratory Transmission Control Protocol (TCP)
   MIL-STD-1778 Traceability Matrix, DRAFT

3. DCEC Protocol Laboratory File Transfer Protocol (FTP)
   MIL-STD-1780 Certification Tests Index, DRAFT

4. DCEC Protocol Laboratory Simple Mail Transfer Protocol (SMTP)
   MIL-STD-1781 Traceability Matrix, DRAFT

5. DCEC Protocol Laboratory TELNET Protocol (TELNET)
   MIL-STD-1782 Certification Tests Index, DRAFT

Online versions are available through anonymous FTP from host SRI-NIC.ARPA or through SERVICE, the NIC's automatic mail service. The pathnames and byte sizes are:

PROTOCOLS:MSTD-1777-TESTS1.DOC	394602 bytes
PROTOCOLS:MSTD-1777-TESTS2.DOC	356791
PROTOCOLS:MSTD-1778-TESTS.DOC	102460
PROTOCOLS:MSTD-1780-TESTS.DOC	62915
PROTOCOLS:MSTD-1781-TESTS.DOC	75244
PROTOCOLS:MSTD-1782-TESTS.DOC	55962

Hardcopies

The DDN NIC can provide hardcopies of the set of documents for $50 (USA and Canada) and $90 overseas to cover the cost of reproduction, postage and handling. Send a written request, accompanied with payment in the form of check, money order, or company purchase order (made payable to *SRI International*) to:

SRI International
DDN Network Information Center
333 Ravenswood Avenue
Menlo Park, CA 94025
800-235-3155 or 415-859-3695

Please include your full name, business address, and zip code. California residents should add 6.5% sales tax to their order total (except military users).

Sending comments

Comments on these documents are to be sent via electronic mail to: COMMENTS@EDN-UNIX.ARPA. Please be sure to indicate to which document or documents your comments refer.

Problem Solving in Large TCP/IP Networks

by Harry Saal, Network General Corp.

Introduction

Isolating problems and improving performance in complex and feature-rich TCP/IP networks is a major challenge in today's LAN environments. To tackle these problems, it's useful to review examples of networking bottlenecks occurring in Ethernet LAN-based TCP/IP systems; similar technology is available for other network hardware, such as the 4 megabit IBM Token Ring. In theory, the same problem-solving methodology could apply to IP running on wide-area-network links, although we have not done so.

What are we looking for?

A considerable amount of work in the protocol analysis field has been focussed on the question of trying to verify protocols and on conformance testing. But establishing the health or problem solving for a fairly large TCP/IP networks has, in our experience, little to do with correctness, per se. Practically all of the successful products commercially available today are *formally* correct. The problems that most users deal with are ones where systems are technically correct, but the source of the problem remains a mystery. Consequently, a protocol verifier, while appealing, actually reveals very little about the causes of anomalous behavior, nor about the differences between varying implementations. The issues of interest are more properly related to particular choices of parameters or algorithms that are being used by the protocols, which are technically within the domain of correctness.

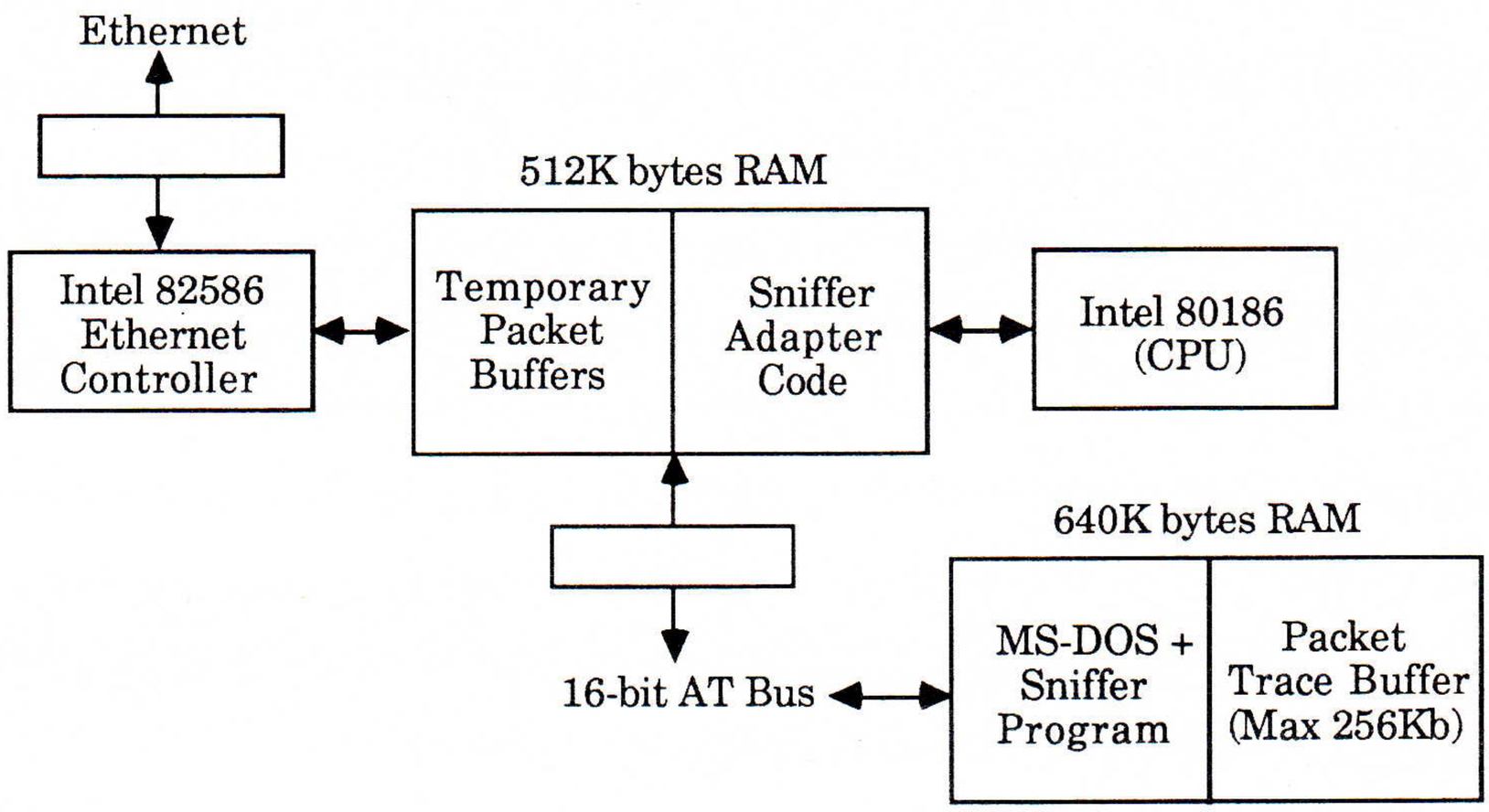
Another point of interest is comparing the quality of different implementations from different vendors. There are significant dissimilarities -- as measured by the number of packets, the number of acknowledgments or the speed of acknowledgment -- that different TCP/IP and FTP packages implement, although their external features and external behavior and even price might seem fairly similar.

TCP/IP evaluation tools

There are a number of packages available which could be used to study TCP/IP. Well known are hardware/software instruments such as the Excelan *LANalyzer*(TM) and Hewlett-Packard *HP 4972A*. There are also software-only offerings such as FTP Software's *LANWatch*(TM) or the public-domain MIT *NetWatch* portion of PC/IP. Each of the above has its strengths and weaknesses, and vary considerably in their ability to filter and accept high rates of data, sophistication of protocol recognition and analysis, human factors, style and methods of data presentation.

The basic tool we have used to observe the performance of TCP/IP on Ethernets is the one we are most familiar with: a product from Network General Corp. called *The Sniffer*(TM). The Sniffer is a hardware/software combination platformed on certain Compaq or Toshiba portable computers. It incorporates a specially modified high-speed Ethernet board that is used to capture and time stamp packets very accurately. At the same time, The Sniffer selects particular packets of interest in real-time by running dynamically compiled Intel 80186 machine code, in order to reduce the total amount of traffic captured.

Once the packets have been captured, they are displayed in a variety of screen formats, selected by the user. The Sniffer attempts to reverse translate or decompile the basic hex format of the packets into a form that a user who isn't a protocol guru can easily understand. Figure 1 shows an overall block diagram of the Sniffer along with a sample screen which shows, in this instance, SMB and NetBIOS frames on top of various other protocols. In this case the Sniffer has recognized that some packets contain protocols as high as the application level, i.e., SMB, which are shown. Other packets are lower-level acknowledgments that are shown as NetBIOS or TCP frames, etc. (This example is taken from the December 1987 NetBIOS interoperability demo).




```
Frame#  Delta    SMB-Server                      PC-User
    80  0.0249                                  SMB C Open \USR2\LANTEST\TEMP1
    81  0.0403   NETB Data, 0 bytes (of 49)
    82  0.0023   SMB R F=0008 Opened
    83  0.0186                                  TCP D=139 S=256 ACK=1760403 WIN=4043
    84  0.0132                                  SMB C F=0008 Read 4096 at 0
    86  0.0155   NETB Data, 0 bytes (of 1072)
    87  0.0036   SMB R OK
    88  0.0015   NETB Data, 48 bytes
    89  0.0283                                  TCP D=139 S=256 ACK=1761479 WIN=2967
    90  0.0318                                  SMB C F=0005 Read 2 at 323658
    91  0.0157   NETB Data, 0 bytes (of 50)
    92  0.0023   SMB R OK
    93  0.0208                                  TCP D=139 S=256 ACK=1761533 WIN=2913
    94  0.0109                                  SMB C F=0005 Read 162 at 327192
    95  0.0157   NETB Data, 0 bytes (of 210)
    96  0.0028   SMB R OK
    97  0.0192                                  TCP D=139 S=256  ACK=1761747 WIN=2699
    98  0.0142                                  SMB C F=0007 Lock 2 at 3623
    99  0.0164   NETB Data, 0 bytes (of 35)
   100  0.0020   SMB R Locked
   101  0.0213                                  TCP D=139 S=256 ACK=1761786 WIN=2660
   102  0.0109                                  SMB C F=0007 Read 2 at 3623
   103  0.0155   NETB Data, 0 bytes (of 50)
```

Figure 1: Block diagram of the Sniffer with sample screen.

continued on next page

Problem Solving in Large TCP/IP Nets *(continued)*

For example, it may not be evident that on an Ethernet the IP protocol actually can be found by The Sniffer under at least four different forms of encapsulation -- all of which the system handles transparently. For instance, IP may be under Ethertype 0800; it may also appear embedded in trailer-style Berkeley packets. It may appear using the SNAP protocol within 802.2-style packets on an 802.3 Ethernet. Lastly, it may appear on the 802.2 service access point assigned specifically to IP. In any case, the user can select IP or higher level protocols (FTP, Telnet, Domain Name System (DNS), etc.) and the system coordinates all the rest of the details.

This knowledge-based approach makes it easy for users to focus on the actual activity in progress rather than immersing themselves in details of the protocol encapsulation.

TCP/IP network problems

We now present several examples of the sorts of problems found on virtually every TCP/IP network we've examined. This is not to say that these particular examples appear on every network, but they are typical of the types of problems that are found on even the best-run TCP/IP networks. Figure 2 shows DNS frames with duplicates appearing. The example shows how domain name service is being used by a typical mailer program. It's clear that many of the names that it's looking up, in fact, are intended to be the same station. The DNS protocol specifies that upper and lower case are not significant, but the mailer program did not consider this and, consequently, sends out many more requests than necessary by not doing the upper/lower case equivalencing. This is a simple opportunity to significantly enhance the behavior and use of this particular protocol, and could reduce the load on a name server by several factors, given the typical requests that it's receiving.

```
Frame#  Delta  Destination    Source
   31  0.0078  [36.28.0.80]   [36.54.0.12]  TCP D=32529 S=23     WIN=4096
   32  1.6780  [36.54.0.12]   [36.54.0.11]  DNS C ID=1 OP=QUERY NAME=Lindy
   33  0.0637  [36.54.0.12]   [36.54.0.11]  DNS C ID=2 OP=QUERY NAME=Lindy
   34  0.0466  [36.54.0.12]   [36.54.0.11]  DNS C ID=3 OP=QUERY NAME=lindy.arpa
   35  0.0306  [10.0.0.52]    [36.54.0.12]  DNS C ID=1104 OP=QUERYNAME=lindy.arpa
   36  0.7620  [36.54.0.12]   [10.0.0.52]   DNS R ID=1104 STAT=Name error NAME=lindy.arpa
   37  0.0400  [36.54.0.12]   [36.54.0.11]  DNS C ID=4 OP=QUERY NAME=Lindy
   38  0.0469  [36.54.0.12]   [36.54.0.11]  DNS C ID=5 OP=QUERY NAME=lindy
   39  0.0534  [36.54.0.12]   [36.54.0.11]  DNS C ID=6 OP=QUERY NAME=lindy.STANFORD.EDU
   40  0.8871  [36.255.255..  [36.54.0.12]  RWHO HOST=Lindy
   41  0.0097  [36.255.255..  [36.54.0.12]  RWHO HOST=Lindy
   42  0.0083  [36.255.255..  [36.54.0.12]  RWHO HOST=Lindy
   43  0.0154  [36.255.255..  [36.54.0.12]  RWHO HOST=Lindy
   44  2.0590  [36.54.0.12]   [36.54.0.11]  DNS C ID=7 OP=QUERY NAME=forsythe.stanford.edu
   45  2.1157  [36.54.0.12]   [36.54.0.11]  TCP D=53 S=1097 ACK=443976706 WIN=4096
   46  1.5739  [36.54.0.12]   [36.54.0.11]  DNS C ID=8 OP=QUERY NAME=score.stanford.edu
   47  0.1334  [36.54.0.12]   [36.54.0.11]  DNS C ID=9 OP=QUERY NAME=score.stanford.edu
   48  1.1544  [36.54.0.12]   [36.54.0.11]  DNS C ID=10 OP=QUERY NAME=forsythe.stanford.edu
   49  0.0794  [36.54.0.12]   [36.54.0.11]  DNS C ID=11 OP=QUERY NAME=score.stanford.edu
   50  0.1052  [36.54.0.12]   [36.54.0.11]  DNS C ID=12 OP=QUERY NAME=Score.Stanford.EDU
   51  0.1463  Broadcast      Lindy IL1     ARP C PA=[36.8.0.46] PRO=IP
```

Figure 2: DNS duplicate requests

Frame#	Delta	Destination	Source	
1		[128.18.4.0]	[128.18.4.2]	RWHO HOST=joyce
2	0.0013	Broadcast	0800200183E8	ARP C PA=[128.18.4.0] PRO=IP
3	0.0003	Broadcast	0800200186FC	ARP C PA=[128.18.4.0] PRO=IP
4	0.0001	Broadcast	080020017CD5	ARP C PA=[128.18.4.0] PRO=IP
5	0.0000	Broadcast	080020018D10	ARP C PA=[128.18.4.0] PRO=IP
6	0.0000	Broadcast	080020018552	ARP C PA=[128.18.4.0] PRO=IP
7	0.0002	Broadcast	080020017F6E	ARP C PA=[128.18.4.0] PRO=IP
8	0.0002	Broadcast	080020013466	ARP C PA=[128.18.4.0] PRO=IP
9	0.0001	Broadcast	08002001A9A2	ARP C PA=[128.18.4.0] PRO=IP
10	0.0001	Broadcast	0800200188AD	ARP C PA=[128.18.4.0] PRO=IP
11	0.0000	Broadcast	08002001715E	ARP C PA=[128.18.4.0] PRO=IP
12	0.0001	Broadcast	08002001A9CD	ARP C PA=[128.18.4.0] PRO=IP
13	0.0001	Broadcast	08002001A2A5	ARP C PA=[128.18.4.0] PRO=IP
14	0.0000	Broadcast	080020018B3B	ARP C PA=[128.18.4.0] PRO=IP
15	0.0000	Broadcast	080020013526	ARP C PA=[128.18.4.0] PRO=IP
16	0.0001	Broadcast	0800200174DC	ARP C PA=[128.18.4.0] PRO=IP
17	0.0001	Broadcast	080020018896	ARP C PA=[128.18.4.0] PRO=IP
18	0.0001	Broadcast	080020018BA8	ARP C PA=[128.18.4.0] PRO=IP
19	0.0000	Broadcast	080020013372	ARP C PA=[128.18.4.0] PRO=IP
20	0.0003	Broadcast	080020017F14	ARP C PA=[128.18.4.0] PRO=IP

Figure 3: Broadcast Storm

Broadcast Storm

The next example represents a common situation that occurs on a large number of networks using Berkeley UNIX systems. Figure 3 shows an ARP broadcast storm. Hosts periodically broadcast to the Ethernet hardware broadcast address a *rwho* packet (describing the list of logged-on users) while including in the packet a specific Internet destination address. Several TCP/IP implementations perform a "service" by attempting to forward this packet, which they have mysteriously found in their input buffers (due to the use of hardware broadcast address). The result: a literal storm of, in this case, approximately 150 ARP requests followed by another 150 ARP replies. This activity often debilitates many Ethernet front-ends, causing total disruption of all services on that Ethernet.

Inefficient TCP

The next example, (Figure 4), is a perfectly legal but not very attractive use of TCP. This particular implementation apparently manages its internal TCP buffers inefficiently and fragments them. Every time the application program attempts to transmit a large block, such as 4,096 bytes of data, it simply picks up whatever bits and pieces of TCP buffers it has, including fragments as small as 1 byte. This generates many times the reasonable number of TCP frames, all of which end up going through gateways. These gateways are typically limited by the number of frames per second they can handle; that is, they can handle just as many 1,500-byte TCP/IP frames as they can handle 1-byte IP frames. Here, again, is an opportunity for major enhancement of this implementation.

continued on next page

Problem Solving in Large TCP/IP Nets *(continued)*

```
Frame#  Delta      Host-1                          Host-2
    93  0.0057  TCP D=1103 S=20    ACK=4667 SEQ=1869 LEN=1195 WIN=4096
    94  0.0041  TCP D=1103 S=20    ACK=4667 SEQ=3064 LEN=1450 WIN=4096
    95  0.0018  TCP D=1103 S=20    ACK=4667 SEQ=4514 LEN=1 WIN=4096
    96  0.1138                                TCP D=20 S=1103 ACK=4515 WIN=2645
    97  0.0021                                TCP D=20 S=1103 ACK=4515 WIN=4096
    98  0.0165  TCP D=1103 S=20    ACK=4667 SEQ=4515 LEN=1450 WIN=4096
    99  0.0049  TCP D=1103 S=20    ACK=4667 SEQ=7160 LEN=1450 WIN=4096
   100  0.0040  TCP D=1103 S=20    ACK=4667 SEQ=5965 LEN=1195 WIN=4096
   101  0.0012  TCP D=1103 S=20    ACK=4667 SEQ=8610 LEN=1 WIN=4096
   102  0.0621                                TCP D=20 S=1103 ACK=5965 WIN=4096
   103  0.0125  TCP D=1103 S=20    ACK=4667 SEQ=8611 LEN=1450 WIN=4096
   104  0.0051                                TCP D=20 S=1103 ACK=8611  WIN=4095
   105  0.0148  TCP D=1103 S=20    ACK=4667 SEQ=10061 LEN=1194 WIN=4096
   106  0.0041  TCP D=1103 S=20    ACK=4667 SEQ=11255 LEN=1450 WIN=4096
   107  0.0012  TCP D=1103 S=20    ACK=4667 SEQ=12705 LEN=1 WIN=4096
   108  0.0225                                TCP D=20 S=1103 ACK=8611 WIN=4096
   109  0.0153                                TCP D=20 S=1103 ACK=10061 WIN=4096
   110  0.0135  TCP D=1103 S=20    ACK=4667 SEQ=12706 LEN=1450 WIN=4096
   111  0.0015                                TCP D=20 S=1103 ACK=11255 WIN=4096
   112  0.0003  TCP D=1103 S=20    ACK=4667 SEQ=14156 LEN=1 WIN=4096
   113  0.0127  TCP D=1103 S=20    ACK=4667 SEQ=14157 LEN=1194 WIN=4096
```

Figure 4: Inefficient TCP

```
Frame#  Delta      Host-1                          Host-2
     1          FTP R PORT=1102 200 Command completed, and okay.<0D><0A>
     2  0.0342                                FTP C PORT=1102 RETR ftptest1<0D><0A>
     3  0.0027                                TCP D=21 S=1102 ACK=5975 WIN=4096
     4  0.0033  TCP D=1102 S=21      ACK=60318652 WIN=80
     5  0.3777  FTP R PORT=1102 150 File status okay, about to connect.<0D><0A>
     6  0.0300                                TCP D=21 S=1102 ACK=6016 WIN=4096
     7  0.5177  TCP D=1103 S=20 SYN SEQ=2108 LEN=0 WIN=4096
     8  0.0279                                TCP D=20 S=1103 SYN ACK=2109 SEQ=4666
                                              LEN=0 WIN=4096
     9  0.0072  TCP D=1103 S=20      ACK=4667 WIN=4096
    10  0.1358  TCP D=1103 S=20      ACK=4667 SEQ=2109 LEN=1450 WIN=4096
    11  0.0060  TCP D=1103 S=20      ACK=4667 SEQ=3559 LEN=1450 WIN=4096
    12  0.0035  TCP D=1103 S=20      ACK=4667 SEQ=5009 LEN=1196 WIN=4096
    13  5.1115  TCP D=1103 S=20      ACK=4667 SEQ=2109 LEN=1450 WIN=4096
    14  0.0054  TCP D=1103 S=20      ACK=4667 SEQ=3559 LEN=1450 WIN=4096
    15  0.0035  TCP D=1103 S=20      ACK=4667 SEQ=5009 LEN=1196 WIN=4096
    16  5.1135  TCP D=1103 S=20      ACK=4667 SEQ=2109 LEN=1450 WIN=4096
    17  0.0040  TCP D=1103 S=20      ACK=4667 SEQ=3559 LEN=1450 WIN=4096
    18  0.0066  TCP D=1103 S=20      ACK=4667 SEQ=5009 LEN=1196 WIN=4096
    19  0.0908                            TCP D=20 S=1103 ACK=6205 WIN=2900
    20  0.0131  TCP D=1103 S=20      ACK=4667 SEQ=6205 LEN=1450 WIN=4096
    21  0.0076  TCP D=1103 S=20      ACK=4667 SEQ=7655 LEN=1450 WIN=4096
```

Figure 5: TCP Retransmissions

Retransmissions

The last example (Figure 5) is one where retransmissions are occurring during an FTP transfer between two hosts. In this case, there are two interesting situations. Firstly, if you look at frame 13, you will observe that there was a retransmission of the data after approximately a 5-second timeout. This timeout may, in fact, be far too long and could be reduced. On the other hand, the *reason* for the timeout is most interesting. It appears as if an acknowledgment was not transmitted by the receiving station, although the frames, shown in the Figure as 10-13, are seen. The direct cause of the problem was that the three frames, 10-13, were transmitted so efficiently, that is to say, so close together, that the receiving host's front end was overloaded and missed the receipt of these frames because the transmitting station was simply "too good". In this case, the proper fix is either to buy a faster front end or slow down the upstream host. Alternatively, simply reducing the retransmission retry timer or reducing the size of the window (to where at most one packet would be sent rather than three) would improve this file transfer operation many times over.

Summary

We have seen in these four examples just how easy it is to make major performance enhancements in TCP/IP networks running on Ethernet. Use of the latest in protocol analysis technology gives the user a new window on the world to resolving problems in large TCP/IP networks. The need for this type of analysis only grows more and more important as interoperability spreads. Interoperability means not only that products from different vendors should work together and play together. It also means that products from different vendors can be compared and contrasted in order to select the best one. Without doubt, this will significantly improve the class or quality of implementations running on large TCP/IP networks.

COS and its Conformance Test Systems

by Chris Rohrer, Member of COS Technical Staff

The Corporation for Open Systems (COS) International is a two-year old consortium of over 60 computer and communications vendors and users. COS was founded to accelerate the introduction of multi-vendor interoperable products on the world market. It uses the seven-layer Open Systems Interconnection (OSI) reference model, Integrated Services Digital Network (ISDN) and related international standards as the foundation on which to achieve its goals.

One of COS's most important functions is the development of conformance tests for OSI products. Already, COS has developed Message Handling System (MHS), File Transfer Access and Management (FTAM), Transport, Internet Protocol and 802.4 LAN test systems. Our success at the *Enterprise Networking Event '88 International*, showcasing COS conformance testing and interoperability of MHS and FTAM products, is a sign of what the future holds for the public: Real OSI products.

Member input

COS's significant progress in just two years can be attributed to the input of its member representatives and the diligent efforts of COS staff. Through their participation in the COS Strategy Forum and its subcommittees, our member representatives prioritize COS's work and determine the direction we will take to meet our goals.

continued on next page

COS and its Conformance Test Systems *(continued)*

Our next area of concentration will be a tester for 802.3 (CSMA/CD) and X.25. After that COS plans to flesh out testing capability at all layers of the OSI stack to provide full conformance testing of OSI products. We are also moving into the realm of ISDN, and exploring OSI network management and interoperability testing.

Conformance testing

To give you an example of what our conformance test systems provide, consider the COS Transport/Internet Protocol Test System. It includes the tester software and the software of a specially configured Transport implementation that runs in a Sun Microsystems 3/160 machine. Using these two processes in a "loop back" mode verifies system integrity upon delivery and installation to a customer, and provides the test system buyer an excellent learning and training tool for the Transport and Internet protocols.

When a test technician runs the conformance tests against an implementation, he executes a suite of tests that will run automatically, producing logs of the test results written to disk for later examination by an engineer. Also, while the tests are executing, a continuous display shows all activity at the layer or layers of interest to quickly determine trouble areas. The user interface is designed to require a short learning curve and to be generally easy to operate.

Conformance testing can be improved by detecting any interworking problems in the field. By feeding field experience in OSI and ISDN product use back into the specification and test definition process, gradual elimination of such problems will take place. The improved testing quality should result in similar product improvement, bringing us ever closer to true multivendor interoperability.

Test components

COS supplies a software distribution tape and manual to its customers when they take delivery of any COS test system. COS suggests using the following hardware to run the software: Sun 3/160 with 280 megabyte disk and 8 megabytes of RAM, LAN interface and/or X.25 interface boards.

Since there are several combinations of OSI protocol layers that can be used to fit a given situation's requirements, COS has designed enough flexibility in its testers to allow them to operate over local area networks and wide area network technology such as X.25.

IP

The Internet Protocol (ISO IP) can be tested in 3 protocol configurations or "stacks": 802.3 (CSMA/CD) and 802.4 (Token Bus) LANs, and over X.25.

TP and MHS

The Transport and MHS protocols can each be tested over 5 stacks:

- Transport class 4 over IP on both 802.3 and 802.4 LANs
- Transport class 4 over IP over X.25, and
- Transport classes 0 and 4 directly over X.25.

FTAM

The FTAM tester will test over 4 stacks:

- Transport class 4 over IP on both 802.3 and 802.4 LANs
- Transport class 4 over IP over X.25, and
- Transport class 4 directly over X.25.

Token Bus

The 802.4 (Token Bus) testing is done with two separate systems, one for the physical layer and another for the MAC sublayer of layer 2.

COS testing software will be made available under license to vendors in their own plants to aid product development. Our in-house facility is open to members and non-members, alike.

Profiles

In order for OSI to be truly realized, communicating end-systems have to do more than conform to OSI protocols as written. This is because these protocols offer great flexibility and may contain many optional classes, subsets, and parameters. Through COS, our members work to agree on precisely stated subsets or "profiles" necessary to support certain OSI functions. COS then develops test systems for these functionally complete, useful profiles.

Conformance testing of systems implementing a specific profile will provide a level of assurance that those systems can exchange data freely. It determines whether a specific implementation conforms to the relevant OSI protocol standards.

Harmonization

Conformance testing, like so many other standards issues, needs to be harmonized on an international level. That's why COS member representatives and staff meet with other standards-related organizations around the world. The Manufacturing Automation Protocol and Technical Office Protocol (MAP/TOP) Users Group, the Standards Promotion Application Group (SPAG) of Europe, and the Promotional Council for OSI (POSI) of Japan have all participated in this venture with COS. We all believe that an internationally unified approach will result in reduced chances for needless duplication of effort, while accelerating the standards process at the same time.

Through our joint efforts, standards bodies will receive views presented by our individual organizations, representing global decisions reached through our combined efforts. This should reduce the time needed to reach decisions for further action by standards bodies, because a prior consensus will have been achieved by users and vendors.

COS Mark

Meanwhile, in September, 1988 we plan to introduce the trial phase of our *COS Mark Program* for OSI products that have been successfully tested against COS OSI protocol specifications. When a product is awarded the COS Mark, product literature or advertising will indicate licensing of the COS Mark. It will be accompanied by documents describing the testing and how the vendor implemented the products. The full program is expected to be implemented in first quarter 1989.

Vendors should find that COS test products reduce product development costs, since any bugs found can be worked out prior to product distribution. Users will realize the savings in their network implementation charges, since the development cost of the products won't be as high as private vendor testing would make them. Another plus for users will be the assurance of consistent testing for competitive products.

If you would like more information on COS test systems and test products, please contact Customer Service Representative Lori Leonard at 703-883-2756.

CONNEXIONS
480 San Antonio Road
Suite 100
Mountain View, CA 94040

FIRST CLASS MAIL
U.S. POSTAGE
PAID
SAN JOSE, CA
PERMIT NO. 1

CONNEXIONS

PUBLISHER Daniel C. Lynch

EDITOR Ole J. Jacobsen

CONNEXIONS

Subscribe to CONNEXIONS

U.S./Canada	$100. for 12 issues/year	$180. for 24 issues/two years	$240. for 36 issues/three years
International	$ 50. additional **per year**	**(Please apply to all of the above.)**	

Name _____ Title _____

Company _____

Address _____

City _____ State _____ Zip _____

Country _____ Telephone ( ) _____

☐ Check enclosed (in U.S. dollars made payable to **CONNEXIONS**).
☐ Charge my ☐ Visa ☐ Master Card Card # _____ Exp. Date _____

Signature _____

Please return this application with payment to:

CONNEXIONS
480 San Antonio Road Suite 100
Mountain View, CA 94040
415-941-3399

Back issues available upon request $10./each
Volume discounts available upon request